Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L1	457	717/140.ccls.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/09/01 16:42
L2	2	717/140.ccls. and (bit or bitwise) and "constant propagation" and precision	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/09/01 16:44
L3	2	717/140.ccls. and "constant propagation" and precision	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/09/01 16:44
L4	2	717/140.ccls. and "constant propagation" and precision	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/09/01 16:44
L5	5	717/140.ccls. and (bit or bitwise) and "constant propagation"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/09/01 16:45
L6	0	717/140.ccls. and ((replac$3 or substitut$3) near2 opertor) and "constant propagation"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/09/01 16:45
S1	1	"bitwise constant propagation" same ((reduced or smaller) near3 type)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/11 18:11
S2	1	"bitwise constant propagation" same (demot$3 or ((reduced or smaller) near3 type))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/11 17:10
S3	2	"20030188299"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/11 17:20

S4	6	"20040019883" or ("6223142" "5960171" "67421808").pn.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/11 17:21
S5	3	"bitwise constant propagation"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/11 17:53
S6	3	(bitwise or bit-wise) near3 "constant propagation"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/11 17:54
S7	0	("6848099").URPN.	USPAT	OR	OFF	2005/04/11 17:55
S8	17	(bitwise or bit-wise or bit) same "constant propagation"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/11 18:12
S9	65240	(demot$3 or reduc$4) same precision	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/11 18:14
S10	34	(demot$3 or reduc$4) same precision and "constant propagation"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/11 18:14
S11	0	(translat$3 or transform$3 or optimiz$5 or optimis$5) same (reduc$3 near2 "size of") and reduc$3 near5 (type or code)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/28 17:28
S12	0	(translat$3 or transform$3 or optimiz$5 or optimis$5) same (reduc$3 near2 "size of") and reduc$3 near5 (type or code)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/28 17:28
S13	2760	(translat$3 or transform$3 or optimiz$5 or optimis$5) same reduc$3 near5 (type or code)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/28 17:29
S14	22	(translat$3 or transform$3 or optimiz$5 or optimis$5) same reduc$4 near5 (type or code) and reduc$4 near3 precision	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/28 17:30

S15	2	(translat$3 or transform$3 or optimiz$5 or optimis$5) same (type near3 demot$3)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/28 17:32
S16	46	(translat$3 or transform$3 or optimiz$5 or optimis$5) same demot$3	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/28 17:35
S17	22	type near3 demot$3	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/04/28 17:35
S18	19	S17 not S16	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/04/28 17:35
S19	0	bitwidth near3 analysis	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/30 10:57
S20	128	bitwidth	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/30 11:12
S21	20	("4766566" | "4972314" | "5175843" | "5197127" | "5550749" | "5581762" | "5606698" | "5619692" | "5666535" | "5668948" | "5729466" | "5742814" | "5870308" | "6026228" | "6192504" | "6216252" | "6237021" | "6421809" | "6463560" | "6505328").PN. OR ("6772399").URPN.	US-PGPUB; USPAT; USOCR	OR	OFF	2005/04/30 11:06
S22	3571	bitwise	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/30 11:12
S23	1491	bitwise and (minimiz$3 or minimal)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/30 11:13

S24	382	bitwise and (minimiz$3 or minimal) and precision	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/30 11:13
S25	280	bitwise and (minimiz$3 or minimal) and precision and (constant or propagation)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/30 11:34
S26	244	bitwise and (minimiz$3 or minimal) and ((demot$3 or reduc$4) near3 (precision or size or width)) and (constant or propagation)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/04/30 11:36
S27	8	(bitwise or "constant propagation" or "bit-wise") and (("size_of" or size or sizeof or "size of") near2 ("op code" or operation or op or operator)) and (replac$3 or substitut$3) near3 ("op code" or operation or op or operator) and (less$2 or greater or smaller or larger) and precision	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/08/02 10:41
S28	0	"operation on a data type"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/08/02 10:39
S29	603	"operation" near3 "data type"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/08/02 10:39
S30	355	"operation" near2 "data type"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/08/02 10:39
S31	0	"operation" and ((on or to or for) near2 "data type")	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/08/02 10:40

S32	1	(bitwise or "constant propagation" or "bit-wise") and (("size_of" or size or sizeof or "size of") and ("op code" or operation or op or operator)) and (replac$3 or substitut$3) near3 ("op code" or operation or op or operator) and (less$2 or greater or smaller or larger) and precision and S30	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/08/02 10:42
S33	1	(bitwise or "constant propagation" or "bit-wise") and (("size_of" or size or sizeof or "size of") and ("op code" or operation or op or operator or expression)) and (replac$3 or substitut$3) near3 ("op code" or operation or op or operator) and (less$2 or greater or smaller or larger) and precision and S30	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/08/02 10:42
S34	9	(bitwise or "constant propagation" or "bit-wise") and (("size_of" or size or sizeof or "size of") and ("op code" or operation or op or operator or expression)) and (less$2 or greater or smaller or larger) and precision and S30	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/08/02 11:15
S35	5	("5991531").URPN.	USPAT	OR	OFF	2005/08/02 10:49
S36	11	("4370709" | "5345580" | "5678032" | "5832292" | "5889983" | "5896522" | "5970241" | "5991531" | "5991870").PN. OR ("6360194"). URPN.	US-PGPUB; USPAT; USOCR	OR	OFF	2005/08/02 11:11
S37	18657	("op code" or operation or op or operator or expression) near5 precision	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/08/02 11:16
S38	506	("op code" or operation or op or operator or expression) near5 (reduc$4 near3 precision)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/08/02 12:54
S39	1	("op code" or operation or op or operator or expression) near5 (reduc$4 near3 precision) near5 (replac$3 or modify$3 or modificaton or substitut$3 or swap$4)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/08/02 11:18

S40	1	("op code" or operation or op or operator or expression) near5 (reduc$4 near3 precision) near5 (replac$4 or modify$3 or modificaton or substitut$3 or swap$4)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/08/02 11:18
S41	7	("op code" or operation or op or operator or expression) near5 (reduc$4 near3 precision) and (bitwise or "constant propagation" or "bit-wise")	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/08/02 12:49
S45	128	("op code" or operation or op or operator or expression) near5 (reduc$4 near3 precision) and (bit or bitwise or "bit-wise" or "constant propagation")	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/08/02 12:55
S46	8	("4153938" | "4864529" | "5151875" | "5426599" | "5444835" | "5457804" | "5561810" | "5625764").PN. OR ("5935197").URPN.	US-PGPUB; USPAT; USOCR	OR	OFF	2005/08/02 13:02

Google Scholar BETA

| bitwise operator OR precision "constant propa| 1990 | - | 2001 | Search |

Ac
Sc
Sc

Scholar Results 1 - 10 of about **25** for **bitwise operator OR precision "constant propagation"**. (0.05

[PS] Generation of Efficient Interprocedural Analyzers with PAG
M Alt, F Martin - SAS, 1995 - rw4.cs.uni-sb.de
... cessfully been tested by generating several analyzers (eg alias analysis, **constant
propagation**) for an ... to be dened on the left side and an **operator** applied to ...
Cited by 78 - View as HTML - Web Search - cs.uni-sb.de - rw4.cs.uni-sb.de - portal.acm.org - all 6
versions »

Compiling SA-C Programs to FPGAs: Performance Results
BA Draper, APW Boehm, J Hammes, WA Najjar, JR ... - ICVS, 2001 - springerlink.com
... FPGAs to form arbitrary bit **precision** logic cir ... positive differencing, majority
thresholding, **bitwise** "and", percentile ... into a single **operator** that passes ...
Cited by 10 - Web Search - cs.ucr.edu - cs.colostate.edu - portal.acm.org - all 6 versions »

[PS] An Efficient Decision Procedure for the Theory of Fixed-Sized Bit-Vectors
D Cyrluk, MO Moeller, H Ruess - CAV, 1997 - kestrel.edu
... Then, the solver for the core bit-vector theory is extended to handle other
bit-vector operations like **bitwise** logical operations, shifting, and arithmetic ...
Cited by 18 - View as HTML - Web Search - daimi.au.dk - csl.sri.com - informatik.uni-ulm.de - all 11
versions »

Bitwidth analysis with application to silicon compilation
M Stephenson, J Babb, S Amarasinghe - ACM SIGPLAN 2000 Conference on Programming Language
Design ..., 2000 - portal.acm.org
... For the **Bitwise** compiler we chose to propagate data- ranges, not only ... most important
applications use arithmetic and will benet from their exact **precision**. ...
Cited by 93 - Web Search - lcs.mit.edu - portal.acm.org - csa.com

A High-Performance Flexible Architecture for Cryptography
RR Taylor, SC Goldstein - CHES, 1999 - springerlink.com
... 5. **Constant propagation** can be performed, reducing the complexity ... combined into a
single **operator** by setting ... operations in cryptography are **bitwise** shifts and ...
Cited by 25 - Web Search - www-2.cs.cmu.edu - madchat.org - inf.pucrs.br - all 11 versions »

[PS] A Compiler for Natural Semantics
M Pettersson - CC, 1996 - ida.liu.se
... by using the match **operator** on the same ... to its CPS representation, mainly **constant
propagation**, copy propagation ... is used to branch to the **bitwise** 'and' of ...
Cited by 8 - View as HTML - Web Search - brics.dk - portal.acm.org - all 9 versions » - Library Search

Marmot: an optimizing compiler for Java
R Fitzgerald, TB Knoblock, E Ruf, B Steensgaard, D ... - Software Practice and Experience, 2000 -
doi.wiley.com
... it encodes boolean operations as **bitwise** integral operations ... transformation-specific
phi-**operator** maintenance is ... representation, reducing the **precision** loss due ...

Cited by 98 - Web Search - sdsc.edu - charlotte.ucsd.edu - cmt.research.microsoft.com - all 17 versions
»

Fast compilation for pipelined reconfigurable fabrics
M Budiu, SC Goldstein - ACM SIGDA INT SYMP FIELD PROGRAM GATE ARRAYS, 1999 -
portal.acm.org
... code elimination, bit-level **constant propagation**, algebraic simplifications ... 8 shows
how a "**bitwise** and" operation ... b are compared with a < **operator**, then to ...
Cited by 33 - Web Search - www-2.cs.cmu.edu - crhc.uiuc.edu - nano.ce.gatech.edu - all 12 versions »

Lattice Frameworks for Multisource and Bidirectional Data Flow Problems
TJ MARLOWE, BG RYDER - ACM Transactions on Programming Languages and Systems, 1995 -
portal.acm.org
... meet **operator**. ... flow frameworks concerning convergence time and solution **precision**
for these ... and Lake's [1993] analysis of **constant propagation** and liveness ...
Web Search - newit.gsu.unibel.by - portal.acm.org

Transformational design of a direction detector for the progressive scan conversion processor
PFA MIDDELHOEK - 1994 - wwwes.cs.utwente.nl
... Obviously the allocation of multiple operations onto a single **operator** in the direction
detector is not possible due to performance constraints. ...
Cited by 6 - View as HTML - Web Search - wwwes.cs.utwente.nl - csa.com

Google ▶

Result Page: 1 2 3   **Next**

| bitwise operator OR precision "const | Search |

Google Home - About Google - About Google Scholar

PARTAL
USPTO

Search: ⊙ The ACM Digital Library   ○ The Guide

+bitwise +constant +propagation +precision operator operation

THE ACM DIGITAL LIBRARY

🖬 Feedback Report a problem Satisfaction su

Published since January 1990 and Published before October
2001
Terms used
bitwise constant propagation precision operator operation

Found **26** of 74

Sort results
by   [relevance ▼]   ◆ Save results to a Binder   Try an Advanced Search
                                             ⓘ Search Tips   Try this search in The ACM Guide
Display
results   [expanded form ▼]   ☐ Open results in a new
                                             window

Results 1 - 20 of 26   Result page: **1** 2   next

Relevance scale ☐ ⊡ ■

1   Compiler transformations for high-performance computing
        David F. Bacon, Susan L. Graham, Oliver J. Sharp
        December 1994 **ACM Computing Surveys (CSUR)**, Volume 26 Issue 4
        Full text available: 🖬 pdf(6.32   Additional Information: full citation, abstract, references, citings,
                              MB)   index terms, review

        In the last three decades a large number of compiler transformations for optimizing programs have
        been implemented. Most optimizations for uniprocessors reduce the number of instructions
        executed by the program using transformations based on the analysis of scalar quantities and data-
        flow techniques. In contrast, optimizations for high-performance superscalar, vector, and parallel
        processors maximize parallelism and memory locality with transformations that rely on tracking
        the properties o ...

        Keywords: compilation, dependence analysis, locality, multiprocessors, optimization, parallelism,
        superscalar processors, vectorization

2   Lattice frameworks for multisource and bidirectional data flow problems
        Stephen P. Masticola, Thomas J. Marlowe, Barbara G. Ryder
        September 1995 **ACM Transactions on Programming Languages and Systems (TOPLAS)**,
                      Volume 17 Issue 5
        Full text available: 🖬 pdf(1.66   Additional Information: full citation, abstract, references, citings,
                              MB)   index terms, review

        Multisource data flow problems involve information which may enter nodes independently
        through different classes of edges. In some cases, dissimilar meet operations appear to be used for
        different types of nodes. These problems include bidirectional and flow-sensitive problems as well
        as many static analyses of concurrent programs with synchronization. K-tuple frameworks, a type
        of standard data flow framework, provide a na ...

        Keywords: data flow analysis, lattice frameworks

3 Query evaluation techniques for large databases
Goetz Graefe
June 1993 **ACM Computing Surveys (CSUR)**, Volume 25 Issue 2
Full text available: ▓ pdf(9.37   Additional Information: full citation, abstract, references, citings,
            MB)   index terms, review

Database management systems will continue to manage large data volumes. Thus, efficient
algorithms for accessing and manipulating large sets and sequences will be required to provide
acceptable performance. The advent of object-oriented and extensible database systems will not
solve this problem. On the contrary, modern data models exacerbate the problem: In order to
manipulate large sets of complex objects as efficiently as today's database systems manipulate
simple records, query-processi ...

Keywords: complex query evaluation plans, dynamic query evaluation plans, extensible database
systems, iterators, object-oriented database systems, operator model of parallelization, parallel
algorithms, relational database systems, set-matching algorithms, sort-hash duality

4 C and tcc: a language and compiler for dynamic code generation
Massimiliano Poletto, Wilson C. Hsieh, Dawson R. Engler, M. Frans Kaashoek
March 1999 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume
            21 Issue 2
Full text available: ▓ pdf(471.68   Additional Information: full citation, abstract, references, citings,
            KB)   index terms, review

Dynamic code generation allows programmers to use run-time information in order to achieve
performance and expressiveness superior to those of static code. The 'C(Tick C) language is a
superset of ANSI C that supports efficient and high-level use of dynamic code generation. 'C
provides dynamic code generation at the level of C expressions and statements and supports the
composition of dynamic code at run time. These features enable programmers to add dynamic
code generation ...

Keywords: ANSI C, compilers, dynamic code generation, dynamic code optimization

5 Composite model-checking: verification with type-specific symbolic representations
Tevfik Bultan, Richard Gerber, Christopher League
January 2000 **ACM Transactions on Software Engineering and Methodology (TOSEM)**,
            Volume 9 Issue 1
Full text available: ▓ pdf(400.17   Additional Information: full citation, abstract, references, citings,
            KB)   index terms

There has been a surge of progress in automated verification methods based on state exploration.
In areas like hardware design, these technologies are rapidly augmenting key phases of testing and
validation. To date, one of the most successful of these methods has been symbolic model-
checking, in which large finite-state machines are encoded into compact data structures such as
Binary Decision Diagrams (BDDs), and are then checked for safety and liveness properties.
However, these technique ...

Keywords: Presburger arithmetic, binary decision diagrams, symbolic model-checking

6   Code reuse in an optimizing compiler
    Ali-Reza Adl-Tabatabai, Thomas Gross, Guei-Yuan Lueh
    October 1996 **ACM SIGPLAN Notices , Proceedings of the 11th ACM SIGPLAN conference on
            Object-oriented programming, systems, languages, and applications**, Volume 31
            Issue 10
    Full text available: pdf(1.97   Additional Information: full citation, abstract, references, citings,
            MB)   index terms

    This paper describes how the cmcc compiler reuses code---both internally (reuse between differen
    modules) and externally (reuse between versions for different target machines). The key to reuse
    are the application frameworks developed for global data-flow analysis, code generation,
    instruction scheduling, and register allocation.The code produced by cmcc is as good as the code
    produced by the native compilers for the MIPS and SPARC, although significantly less resources
    have been spent on cmcc ...

7   Arithmetic coding revisited
    Alistair Moffat, Radford M. Neal, Ian H. Witten
    July 1998 **ACM Transactions on Information Systems (TOIS)**, Volume 16 Issue 3
    Full text available: pdf(487.26   Additional Information: full citation, abstract, references, citings,
            KB)   index terms

    Over the last decade, arithmetic coding has emerged as an important compression tool. It is now
    the method of choice for adaptive coding on myltisymbol alphabets because of its speed, low
    storage requirements, and effectiveness of compression. This article describes a new
    implementation of arithmetic coding that incorporates several improvements over a widely used
    earlier version by Witten, Neal, and Cleary, which has become a de facto standard. These
    improvements include f ...

    Keywords: approximate coding, arithmetic coding, text compression, word-based model

8   Implementation and tests of low-discrepancy sequences
    Paul Bratley, Bennett L. Fox, Harald Niederreiter
    July 1992 **ACM Transactions on Modeling and Computer Simulation (TOMACS)**, Volume 2
            Issue 3
    Full text available: pdf(1.23   Additional Information: full citation, abstract, references, citings,
            MB)   index terms

    Low-discrepancy sequences are used for numerical integration, in simulation, and in related
    applications. Techniques for producing such sequences have been proposed by, among others,
    Halton, Sobol´, Faure, and Niederreiter. Niederreiter's sequences have the best theoretical
    asymptotic properties. The paper describes two ways to implement the latter sequences on a
    computer and discusses the results obtained in various practical tests on particular integrals.

    Keywords: Niederreiter sequences, Quasi-Monte Carlo methods, low-discrepancy sequences,
    quasirandom sequences

9 Essential language el(&agr;)—a reduced expression set language for system programming
   Tan Watanabe, Kazuko Sakuma, Hideyuki Arai, Kohbun Umetani
   January 1991 **ACM SIGPLAN Notices**, Volume 26 Issue 1
   Full text available: pdf(1.08
                    MB)   Additional Information: full citation, abstract, index terms

   Essential Language el(α) is developed as a "Reduced Expression Set Language" which is
   simple and yet satisfies conditions for efficient programming and efficient execution. The
   grammar of el(α) is determined by unifying a machine model applicable to almost all
   computers and a language model with features essential for system programming. Much efforts
   are exerted to make the language machine independent and to eliminate execution-time overhead.
   Its simple syntax drawn in 2 dia ...

10 Parallel logic simulation of VLSI systems
   Mary L. Bailey, Jack V. Briner, Roger D. Chamberlain
   September 1994 **ACM Computing Surveys (CSUR)**, Volume 26 Issue 3
   Full text available: pdf(3.74   Additional Information: full citation, abstract, references, citings,
                    MB)   index terms

   Fast, efficient logic simulators are an essential tool in modern VLSI system design. Logic
   simulation is used extensively for design verification prior to fabrication, and as VLSI systems
   grow in size, the execution time required by simulation is becoming more and more significant.
   Faster logic simulators will have an appreciable economic impact, speeding time to market while
   ensuring more thorough system design testing. One approach to this problem is to utilize parallel
   processing, taking ...

   Keywords: circuit structure, parallel architecture, parallelism, partitioning, synchronization
   algorithm, timing granularity

11 Packet types: abstract specification of network protocol messages
   Peter J. McCann, Satish Chandra
   August 2000 **ACM SIGCOMM Computer Communication Review , Proceedings of the
            conference on Applications, Technologies, Architectures, and Protocols for
            Computer Communication**, Volume 30 Issue 4
   Full text available: pdf(435.48   Additional Information: full citation, abstract, references, index
                    KB)   terms

   In writing networking code, one is often faced with the task of interpreting a raw buffer according
   to a standardized packet format. This is needed, for example, when monitoring network traffic for
   specific kinds of packets, or when unmarshaling an incoming packet for protocol processing. In
   such cases, a programmer typically writes C code that understands the grammar of a packet and
   that also performs any necessary byte-order and alignment adjustments. Because of the
   complexity of certain ...

12 Standards for system-level design: practical reality or solution in search of a question?
   Christopher K. Lennard, Patrick Schaumont, Gjalt de Jong, Anssi Haverinen, Pete Hardee
   January 2000 **Proceedings of the conference on Design, automation and test in Europe**
   Full text available: pdf(74.65
                    KB)
                    Publisher Site   Additional Information: full citation, references, citings, index
                                                              terms

13 Feature-based control of visibility error: a multi-resolution clustering algorithm for global
   illumination
   Fraŋois Sillion, George Drettakis
   September 1995 **Proceedings of the 22nd annual conference on Computer graphics and
                   interactive techniques**
   Full text available: 📄 pdf(489.21   Additional Information: full citation, references, citings, index
                       KB)   terms

   Keywords: clustering, feature-based error metric, global illumination, hierarchical radiosity,
   multi-resolution visibility, progressive multi-gridding, visibility error

14 Intensional polymorphism in type-erasure semantics
   Karl Crary, Stephanie Weirich, Greg Morrisett
   September 1998 **ACM SIGPLAN Notices , Proceedings of the third ACM SIGPLAN
                   international conference on Functional programming,** Volume 34 Issue 1
   Full text available: 📄 pdf(1.27   Additional Information: full citation, abstract, references, citings,
                       MB)   index terms

   Intensional polymorphism, the ability to dispatch to different routines based on types at run time,
   enables a variety of advanced implementation techniques for polymorphic languages, including
   tag-free garbage collection, unboxed function arguments, polymorphic marshalling, and flattened
   data structures. To date, languages that support intensional polymorphism have required a type-
   passing (as opposed to type-erasure) interpretation where types are constructed and passed to
   polymorphic functions ...

15 Bidwidth analysis with application to silicon compilation
   Mark Stephenson, Jonathan Babb, Saman Amarasinghe
   May 2000 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2000 conference on
            Programming language design and implementation,** Volume 35 Issue 5
   Full text available: 📄 pdf(930.97   Additional Information: full citation, abstract, references, citings,
                       KB)   index terms

   This paper introduces Bitwise, a compiler that minimizes the bitwidth the number of bits used to
   represent each operand for both integers and pointers in a program. By propagating 70 static
   information both forward and backward in the program dataflow graph, Bitwise frees the
   programmer from declaring bitwidth invariants in cases where the compiler can determine
   bitwidths automatically. Because loop instructions comprise the bulk of dynamically executed
   instructions, Bitw ...

16 Bridging fault detection in FPGA interconnects using IDDQ
   L. Zhao, D. M. H. Walker, F. Lombardi
   March 1998 **Proceedings of the 1998 ACM/SIGDA sixth international symposium on Field
               programmable gate arrays**
   Full text available: 📄 pdf(1.02   Additional Information: full citation, abstract, references, citings,
                       MB)   index terms

This paper presents a vector generation approach for testing interconnects in configurable (SRAM-based) Field Programmable Gate Arrays (FPGAs). The proposed approach detects bridging faults and is based on quiescent current (IDDQ monitoring. Compared with previous voltage-based methods, IDDQ testing has the advantage of utilizing a small number of programming phases for configuring the FPGA during the test process w ...

17 Session 11: seminumerical algorithms: Implementation of a portable and reproducible parallel pseudorandom number generator
Daniel V. Pryor, Steven A. Cuccaro, Michael Mascagni, M. L. Robinson
November 1994 **Proceedings of the 1994 ACM/IEEE conference on Supercomputing**
Full text available: pdf(808.78 KB)   Additional Information: full citation, abstract, references

We describe in detail the parallel implementation of a family of additive lagged-Fibonacci pseudorandom number generators. The theoretical structure of these generators is exploited to preserve their well-known randomness properties and to provide a parallel system of distinct cycles. The algorithm presented here solves the reproducibility problem for a far larger class of parallel Monte Carlo applications than has been previously possible. In particular, Monte Carlo applications that undergo "s ...

18 The anatomy of the register file in a multiscalar processor
Scott E. Breach, T. N. Vijaykumar, Gurindar S. Sohi
November 1994 **Proceedings of the 27th annual international symposium on Microarchitecture**
Full text available: pdf(1.16 MB)   Additional Information: full citation, abstract, references, citings, index terms

This paper presents the operation of the register file in the Multiscalar architecture. The register file provides the appearance of a logically centralized register file, yet is implemented as physically decentralized register files, queues, and control logic in a Multiscalar processor. We address the key issues of storage, communication, and synchronization required for successful design and discuss the complications that arise in the face of speculation. In particular, the hardware requi ...

19 External memory algorithms and data structures: dealing with massive data
Jeffrey Scott Vitter
June 2001 **ACM Computing Surveys (CSUR),** Volume 33 Issue 2
Full text available: pdf(828.46 KB)   Additional Information: full citation, abstract, references, citings, index terms

Data sets in large applications are often too massive to fit completely inside the computers internal memory. The resulting input/output communication (or I/O) between fast internal memory and slower external memory (such as disks) can be a major performance bottleneck. In this article we survey the state of the art in the design and analysis of external memory (or EM) algorithms and data structures, where the goal is to exploit locality in order to reduce the I/O costs. We consider a varie ...

Keywords: B-tree, I/O, batched, block, disk, dynamic, extendible hashing, external memory, hierarchical memory, multidimensional access methods, multilevel memory, online, out-of-core, secondary storage, sorting

20 Eliminating cache conflict misses through XOR-based placement functions
   Antonio González, Mateo Valero, Nigel Topham, Joan M. Parcerisa
   July 1997 **Proceedings of the 11th international conference on Supercomputing**
   Full text available: pdf(1.21   Additional Information: full citation, references, citings, index
                     MB)   terms

   Keywords: XOR-based placement functions, cache memory, conflict misses

Results 1 - 20 of 26   Result page: **1** 2 next

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2005 ACM
Inc.
Terms of Usage   Privacy Policy   Code of Ethics   Contact Us

Useful downloads: Adobe Acrobat   QuickTime   Windows Media Player   Real Playe

IEEE Xplore®

Welcome United States Patent and Trademark
Office

BROWSE   SEARCH   IEEE XPLORE
GUIDE

Search Results

Results for "((((bitwise constant propagation precision operator operation)
<in>metadata)) <and> (pyr &..."   ✉e-mail
Your search matched **0** documents.
A maximum of **100** results are displayed, **25** to a page, sorted by **Relevance** in
Descending order.

» **Search Options**

View Session History

New Search

Modify Search

| ((((bitwise constant propagation precision operator operation)<in>metadat |

☐ Check to search only within this results set

» **Key**

**Display
Format:**   ⦿ Citation ○ Citation & Abstract

IEEE JNL	IEEE Journal or Magazine
IEE JNL	IEE Journal or Magazine
IEEE CNF	IEEE Conference Proceeding
IEE CNF	IEE Conference Proceeding
IEEE STD	IEEE Standard

No results were found.

Please edit your search criteria and try again. Refer to the Help pages
assistance revising your search.

Help   Contact l
Securi

Indexed by
Inspec®